EXHIBIT B21

🗍 **yulei** / **amk**   `Public`

<> **Code** ⊙ Issues ⅄ Pull requests ▷ Actions ⊞ Projects ⦸ Security ⟋ Insights

⑂ **b6d1333a17** ▾   ⋯

amk / **main** / **drivers** / **pca9535.c**

🐸 **yulei** initial addd macro implementation   ⟲ History

⋊ **1 contributor**

116 lines (98 sloc) │ 2.51 KB   ⋯

```c
/**
 * @file pca9535.c
 */

#include "pca9535.h"
#include "i2c.h"
#include "amk_printf.h"

#ifndef PCA9535_DEBUG
#define PCA9535_DEBUG 0
#endif

#if PCA9535_DEBUG
#define pca9535_debug  amk_printf
#else
#define pca9535_debug(...)
#endif

#define PCA9535_INPUT_PORT0     0x00
#define PCA9535_INPUT_PORT1     0x01

#define PCA9535_OUTPUT_PORT0    0x02
#define PCA9535_OUTPUT_PORT1    0x03

#define PCA9535_POLARITY_PORT0  0x04
#define PCA9535_POLARITY_PORT1  0x05

#define PCA9535_CONF_PORT0      0x06
#define PCA9535_CONF_PORT1      0x07
```

```
30
31    #define TIMEOUT                100
32
33    #ifndef PCA9535_I2C_ID
34    #define PCA9535_I2C_ID      I2C_INSTANCE_1
35    #endif
36
37    static i2c_handle_t i2c_inst;
38
39    void pca9535_init(void)
40    {
41        if (!i2c_inst) {
42            i2c_inst = i2c_init(PCA9535_I2C_ID);
43        }
44    }
45
46    static void write_port(uint8_t p, uint8_t d)
47    {
48        i2c_write_reg(i2c_inst, PCA9535_ADDR, p, &d, 1, TIMEOUT);
49    }
50
51    static uint8_t read_port(uint8_t port)
52    {
53        uint8_t data = 0;
54        i2c_read_reg(i2c_inst, PCA9535_ADDR, port, &data, 1, TIMEOUT);
55        return data;
56    }
57
58    void pca9535_write_config(PCA9535_PORT port, uint8_t data)
59    {
60        switch(port) {
61            case PCA9535_PORT0:
62                write_port(PCA9535_CONF_PORT0, data);
63                break;
64            case PCA9535_PORT1:
65                write_port(PCA9535_CONF_PORT1, data);
66                break;
67            default:
68                pca9535_debug("PCA9535: unknown config port:%d\n", port);
69                break;
70        }
71    }
72
73    void pca9535_write_polarity(PCA9535_PORT port, uint8_t data)
74    {
75        switch(port) {
76            case PCA9535_PORT0:
77                write_port(PCA9535_POLARITY_PORT0, data);
78                break;
```

```
 79          case PCA9535_PORT1:
 80              write_port(PCA9535_POLARITY_PORT1, data);
 81              break;
 82          default:
 83              pca9535_debug("PCA9535: unknown polarity port:%d\n", port);
 84              break;
 85      }
 86  }
 87
 88  void pca9535_write_port(PCA9535_PORT port, uint8_t data)
 89  {
 90      switch(port) {
 91          case PCA9535_PORT0:
 92              write_port(PCA9535_OUTPUT_PORT0, data);
 93              break;
 94          case PCA9535_PORT1:
 95              write_port(PCA9535_OUTPUT_PORT1, data);
 96              break;
 97          default:
 98              pca9535_debug("PCA9535: unknown output port:%d\n", port);
 99              break;
100      }
101  }
102
103  uint8_t pca9535_read_port(PCA9535_PORT port)
104  {
105      switch(port) {
106          case PCA9535_PORT0:
107              return read_port(PCA9535_INPUT_PORT0);
108          case PCA9535_PORT1:
109              return read_port(PCA9535_INPUT_PORT1);
110          default:
111              pca9535_debug("PCA9535: unknown input port:%d\n", port);
112              break;
113      }
114
115      return 0;
116  }
```